

# Website Blocker



מגיש: יותם היבשמן רכניץ

ת.ז: 21479907

תיכון ע"ש יגאל אלון רמת השרון

מורים: עידן פלדברג

תאריך הגשה: 20/5

שם העבודה: חוסם אתרים – website blocker



## תוכן עניינים

6.....	מבוא
6.....	ייזום
6.....	תיאור ראשוני של המערכת
6.....	הגדרת הלקוח
6.....	הגדרת יעדים
6.....	בעיות תועלת וחסכונות
6.....	סקירת פתרונות קיימים
6.....	סקירת טכנולוגיות הפרויקט
7.....	תיחום הפרויקט
7.....	פירוט תיאור המערכת
7.....	תיאור מפורט של המערכת
7.....	פירוט היכולות לכל סוג משתמש
7.....	פירוט בדיקות
8.....	תכנון וניהול לוח זמנים
8.....	ניהול סיכונים והדרכים להתמודד איתם
9.....	תיאור תחום הידע
9.....	מסך הרשמה
9.....	מסך כניסה
10.....	מסך ניהול חסימות
10.....	מסך מחיקת משתמש
11.....	ארכיטקטורה של הפרויקט
11.....	תיאור חומרה
11.....	תיאור הטכנולוגיה הרלוונטית
12.....	תיאור זרימת המידע
12.....	דף הרשמה
13.....	דף כניסה

14	דף חסימת אתרים
15	דף מחיקת משתמשים
16	תיאור האלגוריתמים המרכזיים בפרויקט
17	תיאור סביבת הפיתוח
17	תיאור פרוטוקולי התקשורת
17	UDP
18	TCP
20	SSL
23	http
26	https
27	תיאור מסכי מערכת
27	מסך הרשמה
28	מסך כניסה
29	מסך חסימת אתרים
30	מסך מחיקת משתמש
31	היררכיית דפים
31	תיאור מבני הנתונים
31	מסד הנתונים הראשי (SQLITE)
31	קובץ HOST
32	סקירת חולשות ואיומים
32	פיצוח סיסמא
32	SQL INJECTION
32	DDOS
32	ראיית המידע שעובר
32	הזרקת קוד
33	מימוש הפרויקט
33	מודולים מיובאים

33	..... Threading
33	..... re
33	..... socket
33	..... uuid
33	..... typing
33	..... ctypes
33	..... sys
34	..... netifaces
34	..... openssl
34	..... sqlite3
34	..... tkinter
34	..... logging
34	..... SSL
34	..... HASHLIB
35	..... מחלקות ומודלים כתובים
35	..... Computer
35	..... HostClient
36	..... Multisocket
37	..... SQLCLIENT
38	..... broadcast
38	..... GUIClient
40	..... קוד לבעיות אלגוריתמיות
40	..... מיון כתובות IP
41	..... התמודדות עם תשובות BROADCAST
42	..... התמודדות עם הודעות של CLIENTS
43	..... הירשמות לאפליקציה
43	..... מסמך בדיקות

43	..... wireshark
44	..... DB BROWSER
44	..... האם קובץ HOST משתנה
44	..... לבדוק האם החסימה פועלת
44	..... LOGS מסמך
46	..... מדריך למשתמש
46	..... עץ קבצים
47	..... התקנת מערכת
47	..... סביבת עבודה
47	..... הכלים הנדרשים
47	..... רשת
48	..... מדריך לניווט חלונות למשתמש
49	..... רפלקציה
50	..... ביבליוגרפיה

## מבוא

## ייזום

## תיאור ראשוני של המערכת

הפרויקט נודע כדי לחסום אתרים על פי DOMAIN שלהם ולחסום את זה בכל האתרים בהם נמצאה המוצר באותו רשת, בחרתי בפרויקט הזה כי גיליתי על הקובץ HOST שמאפשר לך להגדיר באופן סטטי את הכתובת IP שלה DOMAIN. התאגרים שאני צופה לי בפרויקט הם שיהיה לי קשה לסכרן שכל המחשבים יחסמו את אותם אתרים כל הזמן, ושמירה על המידע המעובר כדי למנוע לפגוע במידע שעובר ובמידע שנשמר במחשב.

## הגדרת הלקוח

המערכת מיועדת להורים שרוצים להגן על ילדיהם מפני אתרים עם תכנים שלא מתאימים להם בנוסף הוא מאפשר לתת עונש ליליד באמצעות חסימת אתרים שבהם הוא משחק.

## הגדרת יעדים

המטרות המרכזיות בפרויקט היא לחסום אתרים על פי רצונו של ההורה, בנוסף המטרות המשניות בפרויקט זה לסכרן את כל המחשבים שמחוברים כך שכולם יחסמו את אותם אתרים, ליצור ממשק משתמש נוח ולהגן על פרטי משתמש.

## בעיות תועלת וחסכונות

הבעיה המרכזית שהמערכת מנסה לפתור היא שהורים אינם יכולים לפקח איפה ילדיהם גולשים באינטרנט וכדי למנוע מצבים שבהם ילדיהם נמצאים באתר שהם לא היו רוצים שהם יהיו בו. המערכת תספק את היכולת למנוע את הגישה לאותם אתרים.

## סקירת פתרונות קיימים

פתרונות אחרים קיימים בכמה צורות כאשר העיקרית היא לנתר את ולחסום את האתרים ישירות דרך הנתב כמו לדוגמה סינון תכנים של בזק. קיימים עוד סוגים שבהם ניתן לחסום כגון חסימה באמצעות שינוי חוקי FIREWALL ועוד.

## סקירת טכנולוגיות הפרויקט

בכדי למנוע גישה לאותם אתרים אני מסתמך על קובץ HOST שקיים רק במערכת ההפעלה WINDOWS וכי על מנת שהקוד יעבוד כמו שצריך חובה להריץ אותו רק על WINDOWS, בנוסף לכך על מנת לקבל גישה לקובץ על המשתמש לאשר קבלת גישה מנהל למחשב.

## תיחום הפרויקט

התחומים בהם הפרויקט עוסק הינם: רשתות כאשר בעיקר יש דגש על הפרוטוקולים הבאים: TCP, DNS ו UDP. הפרויקט מתעסק מעט במערכות הפעלה כאשר צריך לקבל גישה על מנת לנהל את הקובץ HOST וקבלת גישות מנהל.

## פירוט תיאור המערכת

### תיאור מפורט של המערכת

המערכת חוסמת אתרים באמצעות קובץ HOST הנמצא בכל מחשב שעליו קיימת מערכת ההפעלה WINDOWS. המערכת בכל מחשב מתקשרת אם שאר המערכות בחשבים אחרים על מנת לגרום לכך שכולם יחסמו את אותם אתרים. כאשר מתקינים את המערכת בפעם הראשונה צריך להכניס שם משתמש וסיסמא על מנת שרק הבן אדם שהתקין את המערכת יוכל לשנות אילו אתרים המערכת חוסמת. ובכדי לעשות זאת המערכת מנהל בסיס נתונים יחסי שמאפשר שמירת מידע לאורך זמן. הסיסמאות מאובטחות באמצעות פונקציית SHA-256. המערכת מציגה את האופציות השונות שקיימות למשתמש באמצעות ממשק גרפי ייחודי.

## פירוט היכולות לכל סוג משתמש

### מנהל מערכת

יכול לעשות הכול לשנות את החסימות להוסיף חסימות ולמחוק את המשתמש

### משתמש רגיל

בלי שם המשתמש והסיסמא של המנהל הוא לא יכול לעשות כלום בנוגע לאפליקציה והוא יכול לראות רק את דף הכניסה ולא מעבר.

## פירוט בדיקות

הבדיקות שיהיו במערכת הם: בדיקה האם הפקטות עוברות כמו שצריך, האם מסד הנתונים משתנה, האם קובץ HOST משתנה, האם החסימה אכן עובדת, האם קובץ ה LOGS נכון.

## תכנון וניהול לוח זמנים

היעדים להם צריך להיות מוענים עם הפרויקט היו:

8/5 תאריך הגשת ספר פרויקט.

20/5 תאריך הגשת הפרויקט.

כאשר התחלתי לעבוד על הפרויקט התלתי בכך שפתחתי קובץ טקסט והתחלתי לרושם איך הייתי רוצה שהפרויקט יראה אחר כך רשמתי מה עלי ללמוד על מנת לבצע את הפרויקט. ברגע שהבנתי מה עלי ללמוד התחלתי ללמוד ולהתנסות באותם נושאים כאשר ייצרתי כל מיני מיני-פרויקטים שמבוססים רק על אותם ספריות שעלי ללמוד. אחר כך תכננתי לוח זמנים כאשר התחלתי לעבוד על הקוד של הפרויקט בתחילת חופשת פסח.

על מנת לתכנן לוח זמנים נכון חילקתי את הפרויקט לשני חלקים הראשון הקוד והשני הספר כאשר את הקוד חילקתי לעוד כמה חלקים.

הינה הלוח זמנים:

תקשורת בין המחשבים – 15/4-18/4

הצפנת התקשורת – 19/4

ניהול מסד הנתונים – 20/4-22/4

ניהול קובץ ה HOST – 23/4

יצירת ממשק משתמש בסיסי – 24/4-28/4

חיבור כל החלקים יחד – 29/4-30/4

תיקון באגים – 31/4-1/5

כתיבת ספר פרויקט – 2/5 – 8/5

אם צריך לשפר את נראות ממשק המשתמש ותיקון באגים שצצים – 9/5-20/5

## ניהול סיכונים והדרכים להתמודד איתם

הסיכונים העיקריים בפרויקט הם בשמירת שמות משתמשים וסיסמאות ועל מנת להתמודד בסיכון של לקיחת הסיסמאות ובכך להצליח לשנות את האתרים הנחסמים על מנת להתמודד עם זה אני



מצפין את הסיסמא באמצעות SHA-256 ובכך גורם לאופציה שדרך מסד הנתונים יגלו את הסיסמא לבלתי אפשרית.

עוד סיכון קיים בשינוי המידע העובר על מנת להתמודד עם זה כל המידע העובר הנוגע למסד הנתונים מוצפן באמצעות פרוטוקול SSL.

## תיאור תחום הידע

### מסך הרשמה

מהות: רישום משתמש חדש במערכת

אוסף יכולות נדרשות:

ממשק משתמש – מסך הרשמה

קליטת נתונים

בדיקת תיקנות של אותם נתונים האם המשתמש קיים והאם הסיסמא מתאימה למגבלות ואם לא הצגת תגובה מתאימה

הצפנת הסיסמא

שליחת הנתונים לכל שאר המחשבים בצורה מאובטחת.

הצגת מסך הכניסה למשתמש

אובייקטים נחוצים: ממשק משתמש, הצפנה, תקשורת, בסיס נתונים

### מסך כניסה

מהות: כניסת משתמש

אוסף יכולות נדרשות:

ממשק משתמש – מסך כניסה

קליטת נתונים

הצפנת הסיסמא

בדיקה במסד הנתונים האם השם משתמש והסיסמא תואמים ואם הם לא הצגת תגובה מתאימה

ממשק משתמש - הצגת מסך ניהול החסימות

אובייקטים נחוצים: ממשק משתמש, הצפנה, בסיס נתונים

---

## מסך ניהול חסימות

מהות: ניהול חסימת האתרים

אוסף יכולות נדרשות:

ממשק משתמש – הצגת מסך ניהול חסימות

קליטת ה-DOMAIN מהמשתמש

מחיקת DOMIAN

הוספת DOMAIN

אם נוסף או נמחק DOMAIN לעדכן את שאר המחשבים על כך.

סכרון מסד הנתונים של שאר המחשבים

ממשק משתמש – הצגת מסך מחיקת משתמש

אובייקטים נחוצים: ממשק משתמש, הצפנה, תקשורת, בסיס נתונים, קובץ HOST.

---

## מסך מחיקת משתמש

מהות : מחיקת משתמש

אוסף יכולות נדרשות:

ממשק משתמש -הצגת מסך מחיקת משתמש

קליטת סיסמא מהמשתמש

הצפנת הסיסמא מהמשתמש

בדיקת הסיסמא במסד הנתונים

מחיקת המשתמש

עדכון שאר המחשבים על כך שמשתמש נמחק

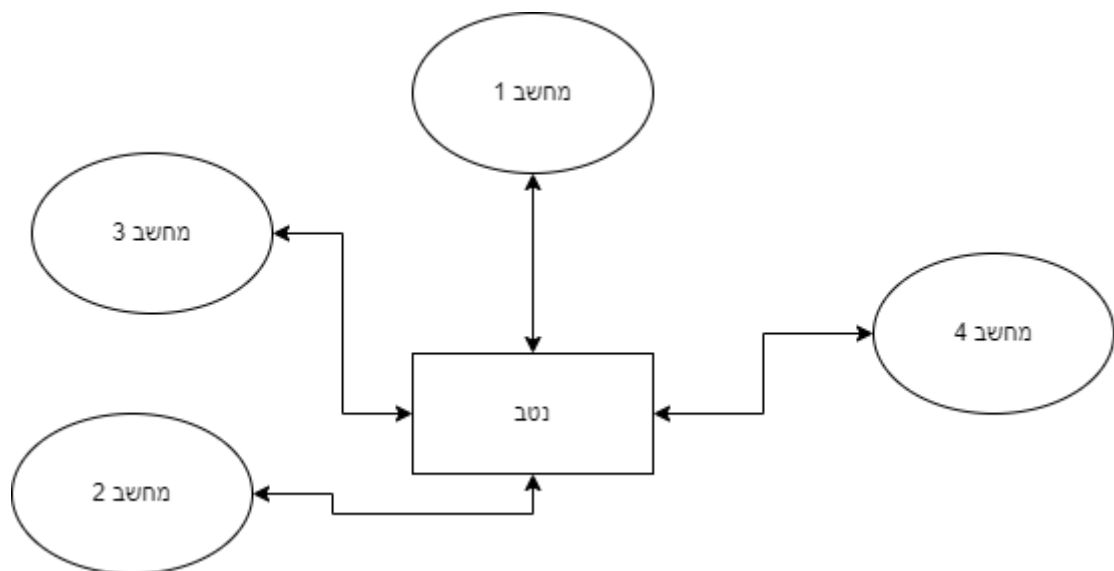
ממשק משתמש העברה למסך הרשמה או כניסה בהתאם למסד הנתונים

אובייקטים נחוצים: ממשק משתמש, הצפנה, תקשורת, בסיס נתונים

### ארכיטקטורה של הפרויקט

### תיאור חומרה

הרשת מורכבת מנתב שעליו מחוברים כל המחשבים כאשר כל מחשב מספק כשרת וכלקוח בו זמנית כאשר המידע עובר דרך הנתב אל כל שאר המחשבים.

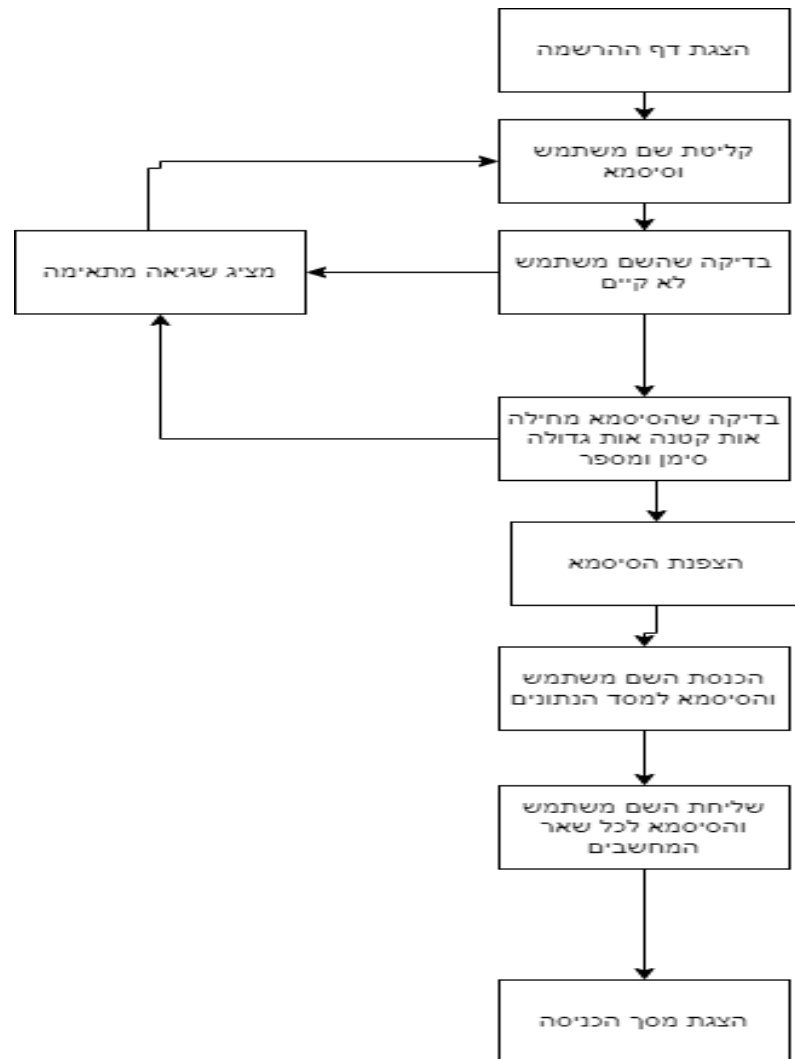


### תיאור הטכנולוגיה הרלוונטית

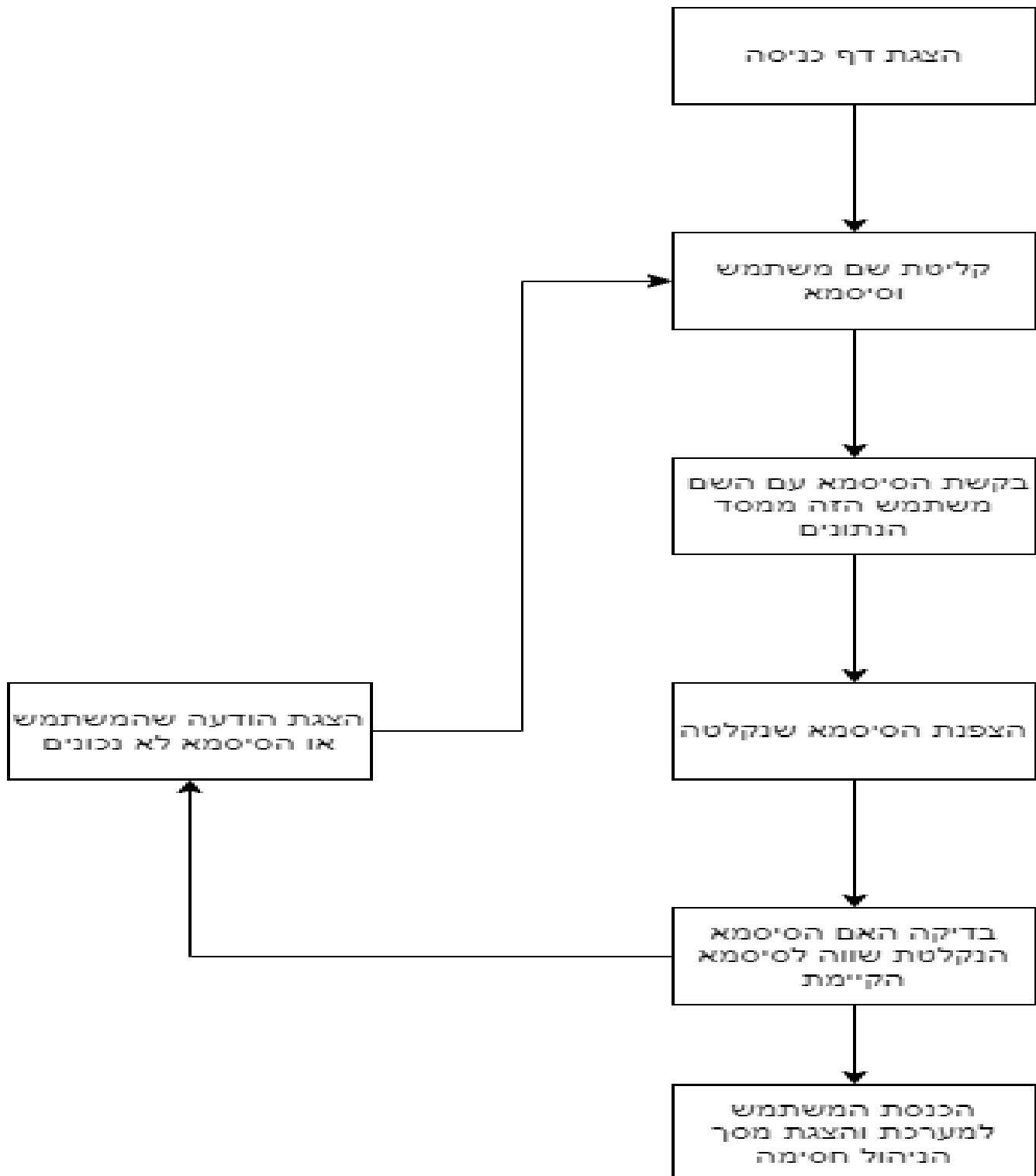
בפרויקט אני משתמש בשתי שפות תכנות: פייתון ו SQL, ובמערכת ההפעלה WINDOWS, כאשר הפרוטוקולים שבהם אני משתמש הם TCP,UDP,SSL,DNS ו HTTP.

## תיאור זרימת המידע

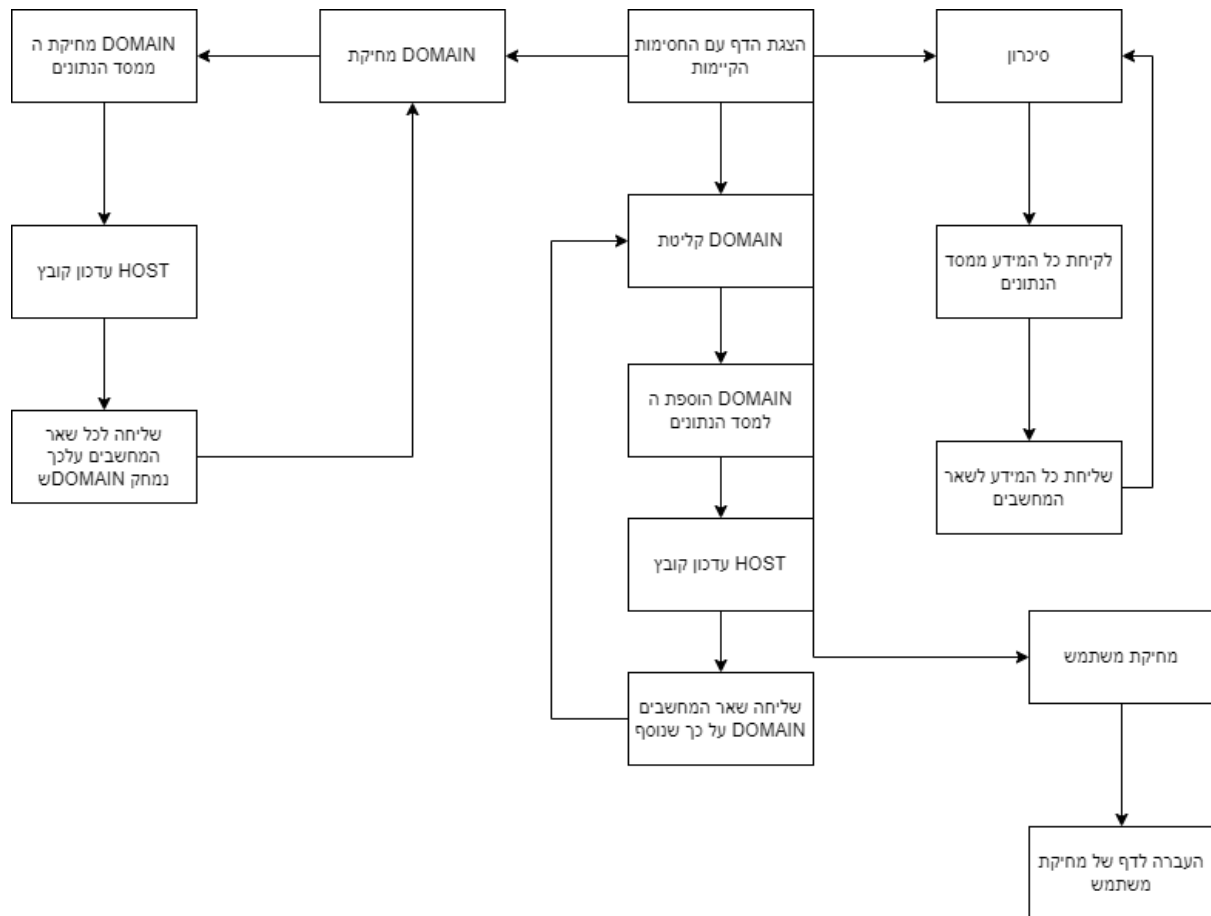
### דף הרשמה



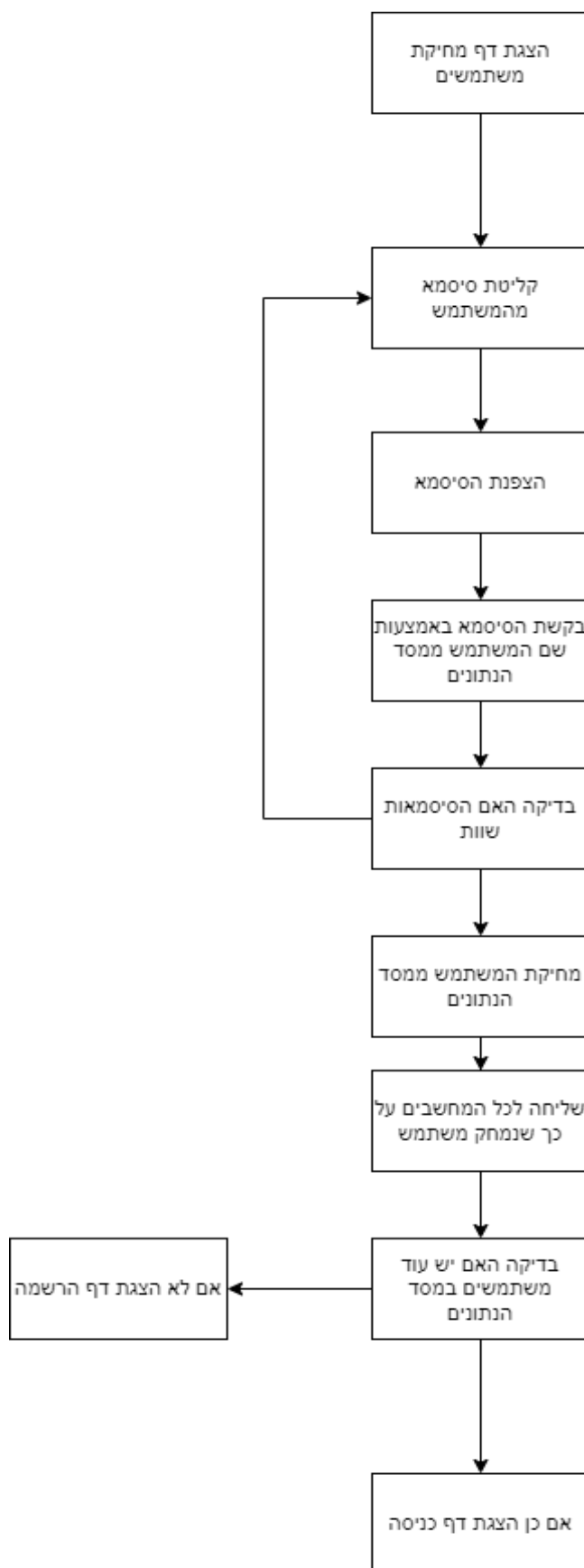
## דף כניסה



## דף חסימת אתרים



## דף מחיקת משתמשים



## תיאור האלגוריתמים המרכזיים בפרויקט

### מיון כתובות IP:

כדי לדעת את ה IP של המחשב אני מקבל רשימה של מילונים שבהם נמצאים ה IP של המחשב ה SUBNET-MASK וכתובת ה BROADCAST של ה SUBNET אבל כדי לדעת מה ה IP האמיתי של הפרויקט גיליתי שזה תמיד ה IP השני בסדר ולכן עלי למיין את הרשימה על פי סדר ה IP.

על מנת למיין את הרשימה יש לעבור על כל IP ולבדוק האם הוא הקטן ביותר עושים זאת באמצעות לחלק את ה IP לארבע חלקים כאשר מפצלים אותם לפי הנקודה ואז ממיינים אותם לפי ששל מי החלק השמאלי קטן יותר הוא קטן יותר מהאחד שמשווים מולו ואם הם שווים עוברים לחלק הבא וכך הלאה.

הסיבה שעשיתי את אלגוריתם מיון זה הוא כי רציתי משהו פשוט שייתן לי את ה IP הנכון אופציות אחרות היו למיין את ה IP על פי אלגוריתם מיון מהר יותר אך בסדר גודל של ה IP שבהם יש במחשב ההבדל הוא בכמה אלפיות השנייה ולכן זה לא מאוד משנה.

### להתמודד עם תשובות BROADCAST

על מנת להוסיף את המחשבים שאליהם מותקנת האפליקציה כל מחשב שולח כל 5 שניות פקטת BROADCAST של UDP שבה נשלח "who is up" וכל שאר המחשבים שקולטים זאת שולחים לאותו מחשב תשובה עם השם של המחשב כתובת ה IP ה SUBNET MASK, PORT, ו כתובת ה MAC. תוך כדי במחלקה הנקראת MultiSocket יש מילון שמכיל את ה IP של המחשב ששלח את ההודעה שביקשה את המידע הנ"ל כמפתח והערך היינו האובייקט של המחשב. כאשר האובייקט של המחשב כרגע מכיל רק את ה IP שלו.

כאשר המחשב ששלח את ה BROADCAST הראשון רואה את ה BROADCAST השני הוא מוסיף את האובייקט של המחשב שלו למילון שלו ופותח ערוץ תקשורת עם המחשב השני.

על מנת שלמחשב השני יהיה את ערוץ התקשורת של הסרבר של המחשב השני קורה אותו תהליך רק ששני המחשבים מתחלפים בתפקיד.

### להתמודד עם הודעות של קליינטים

על מנת לסכרן את מסד הנתונים בין כל המחשבים אני שולח בקשות לשאר המחשבים דרך ה ערוצי תקשורת שפתוחים איתם ואני שולח שלושה סוגי הודעות הוספת משתמש, מחיקת משתמש, הוספת כתובת, מחיקת כתובת. על מנת למנוע זיוף של הודעות וקריאת פרטים אישיים כל הערוצי



תקשורת מוצפנים באמצעות SSL כאשר בכל פועלה כאשר ערוץ התקשורת מזהה את אחת מסוגי ההודעות הללו הוא גם משנה את מסד הנתונים.

### הרשמות לאפליקציה

על מנת להירשם לאפליקציה יש להביא שם משתמש וסיסמא כאשר השם משתמש לא יוכל להיות בשימוש כבר וכאשר הסיסמא צריכה להיות עם לפחות עם אות גדולה אות קטנה מספר וסמל כאשר כל הסיסמא חייבת להיות בגודל של מינימום 8 תווים.

### תיאור סביבת הפיתוח

הכלים הדרושים על מנת לפתח את האפליקציה הם פייטון, SQLi כאשר הסביבות בהם אני השתמשתי הם visual studio code ו pycharm.

### תיאור פרוטוקולי התקשורת

#### UDP

UDP הינו פרוטוקול תקשורת הנמצא בשכבה הרביעית כאשר הוא מאפשר מעבר נתונים דרך כתובות IP ושיוכם לאפליקציות דרך פורט שמשוך לאפליקציה. פרוטוקול זה מספק העבר נתונים מהירה אך לא אמינה כאשר אפשר להשתמש בו גם להעברת בקשות BROADCAST כאשר כתובת היעד בהם היא לא של מחשב ספציפי אלה של כל המחשבים וכל המחשבים יכולים לראות זאת.

למרות שפרוטוקול זה אינו אמין הוא מספק בדיקה מסוימת במסגרת מספר מיוחד הנקרא checksum כאשר פונקציית ה Checksum-מבצעת את החישוב הבא: חילוק ההודעה לקטעים של 16 סיביות, סכימת כל הקטעים והוספת השארית. לאחר מכן התוצאה נשללת לפי ייצוג One's Complement - כל ביט 0 מוחלף ב-1 ולהפך.

חלוקת הבתים בפרוטוקול זה היא לפי הדברים הבאים:

פורט המקור (16 סיביות) - שדה המכיל את מספר הפורט במחשב המקור. שדה אופציונלי, מאחר שפרוטוקול UDP לא מקיים בהכרח תקשורת דו כיוונית. במקרה ואינו בשימוש השדה יכיל אפסים.

פורט היעד (16 סיביות) - שדה המכיל את מספר הפורט במחשב היעד.

אורך החבילה (16 סיביות) - שדה המכיל את אורך פתיח ה UDP והמידע ביחידות של בתים. יש להבדיל משדה "אורך החבילה" שבפתיח של חבילת IPv4 שמכיל את אורך החבילה כולה, בעוד ש"אורך החבילה" בפתיח ה UDP לא מכיל את פתיח פקטת ה IPv4, שגודלו נע בין 20 ל 60 בתים.

סיכום ביקורת (16 סיביות) - checksum, מספר האימות של הפתיח והנתונים. שדה אופציונלי, שבמידה ולא ימומש יכיל רשומה מלאה ב-0 בינארי.

סוגי ההודעות שנשלחות דרך פרוטוקול זה הם:

1. "who is up" – על מנת לדעת אילו מחשבים האפליקציה רצה אליהם כאשר התשובה שמקבים מהמחשבים האחרים היא על פי הפורמט הבא.
2. "ip, subnet mask, mac, port, computer name, up" בעצם ככה נשלח כל המידע הרלוונטי על המחשב הזה. וככה הוא נשלח לכל המחשבים והם יכולים להוסיף אותם ככה.

## TCP

TCP הינו פרוטוקול תקשורת הנמצא בשכבה הרביעית כאשר הוא מאפשר מעבר נתונים דרך כתובות IP ושיוכם לאפליקציות דרך פורט שמשויך לאפליקציה. אך כאשר פותחים ערוץ תקשורת עם מחשב מסוים נוצר תהליך לחיצת יד משולשת כאשר המחשב שיוזם את פתיחת התקשורת שולח למחשב בקשת SYN שמטרת להגיד למחשב השני שאני רוצה לסכך איתו את החיבור, המחשב השני עונה ACK SYN כאשר זה אומר שהמחשב השני קיבל את ההודעה ומסוכך איתך והמחשב הראשון שולח בסוף ACK כדי לסיים את לחיצת היד ולהבהיר למחשב שהכל הגיע. מכאן כל פעם שמידע עובר על מנת לוודא שהוא הגיע כמו שצריך המחשב שמקבל את ההודעה שולח ACK למחשב ששלח את ההודעה.

בנוסף לכך כדי להבטיח שהמידע עובר כמו שצריך אם לא מתקבלת תושבת ACK הפקטה נשלחת שוב עד שיש TIMEOUT שנקבע מראש על פי מערכת ההפעלה או על פי מי שפתח את ערוץ התקשורת.

עוד אמצעים לוודא שהמידע עובר כמו שצריך הם:

- מספור הפקטות כאשר מפצלים אותם כדי לוודא שהמידע מחובר בחזרה בצורה הנכונה.
- Checksum כמו ב UDP.

חלוקת הבתים בפרוטוקול זה הוא בצורה הבאה:

פורט המקור (16 סיביות) - שדה המכיל את מספר הפורט במחשב המקור.

פורט היעד (16 סיביות) - שדה המכיל את מספר הפורט במחשב היעד.

מספר סידורי (32 סיביות) - נבחין בין שני מקרים:

דגל SYN נושא ערך "1" (החבילה היא חלק מהקמת הקשר) - המספר בשדה מכיל את מספר החבילה שממנו יתחילו למנות (המספר הסידורי של חבילת המידע הראשונה יהיה למעשה המספר הזה + 1).

דגל SYN נושא ערך "0" (החבילה היא חלק מרצף התקשורת בין המחשבים) - המספר בשדה מכיל את מספר החבילה הנוכחית ביחס לתקשורת הפתוחה בין שני המחשבים (TCP session).

מספר אישור (32 סיביות) (ACK) - שדה מכיל את המספר הסידורי של ההודעה הבאה לה מצפה התחנה ומהווה אישור קבלה על כל ההודעות בעלות מספר סידורי הקטן ממנו. המידע בשדה תקף רק כאשר דגל ACK נושא ערך "1".

היסט המידע / אורך הפתיח (4 סיביות) - שדה זה מכיל את גודל הפתיח (ביחידות של 32 סיביות) של החבילה הנוכחית שהוא גם ההיסט מתחילת החבילה עד לתחילת המידע.

שמור (3 סיביות) - שדה השמור לשימוש עתידי (אמור להכיל אפסים).

דגלים (9 סיביות)

גודל חלון הקליטה (16 סיביות) - מספר הבתים אותם יכול המחשב לקלוט החל ממספר ההודעה שצוין בשדה מספר האישור. (ACK)

סיכום ביקורת (16 סיביות), checksum, - מספר האימות של הפתיח וחלק מהפתיח של שכבת ה-IP.

היסט למידע דחוף (16 סיביות) - אם דגל URG נושא ערך "1", הערך בשדה זה מציין את ההיסט למיקום מידע דחוף (לרוב אינו בשימוש).

אפשרויות שונות (0–40 בתים) - שדה אופציונלי שיכול להכיל בין היתר את גודל הסגמנט המרבי (Maximum Segment Size, MSS) לשימוש בפרוטוקול. גודל השדה נקבע על פי שדה "אורך הפתיח", אם האורך המצוין אינו גדול מ-5 שדה זה כלל לא נמצא.

ריפוד אפסים (0–4 בתים) - שדה שתפקידו למלא את הפתיח ב-"0" על מנת להגיע לכפולה שלמה של 32 סיביות.

ההודעות שנשלחות דרך פרוטוקול זה הם:

"still alive?" – לוודא שערוץ התקשורת עדיין פתוח ואין בעיה אתו.

"add domain {domain}" – להוסיף את הכתובת הנתונה למסד הנתונים

"remove domain {domain}" – למחוק את הכתובת הנתונה ממסד הנתונים

"add user {username} {encrypted password}" – להוסיף משתמש למסד הנתונים

"remove user {username}" - למחוק משתמש ממסד הנתונים

## SSL

SSL/TLS הוא פרוטוקול ורסטילי שמטרתו אבטחת שיחת שרת/לקוח בשיטות קריפטוגרפיות חזקות והוא אמור למנוע ציתות, זיוף, או חבלה (שינוי זדוני) של המידע העובר בין השרת והלקוח. מאפשר חיבור אנונימי, אימות שרת (חד-צדדי) או אימות דו-צדדי, תוך שמירה על דיסקרטיות ושלמות המסרים. שלוש נקודות עיקריות שהפרוטוקול אמור לתת להן מענה הן:

- פרטיות -המושגת באמצעות הצפנה סימטרית.
- אימות -המושג באמצעות תעודת מפתח ציבורי.
- אותנטיות -המושגת באמצעות קוד אימות מסרים.

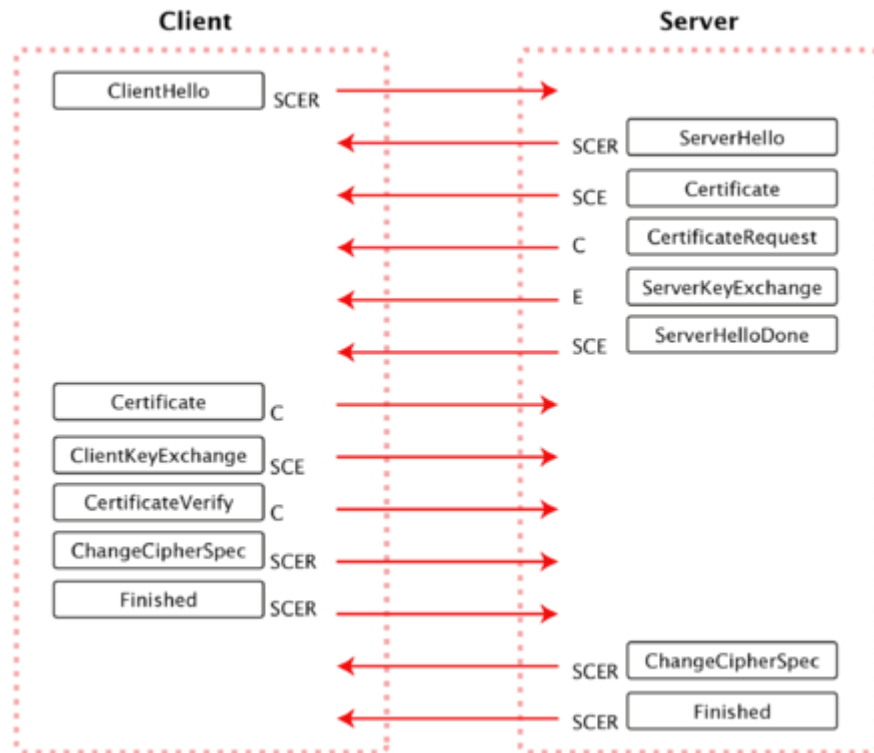
אני משתמש בפרוטוקול זה רק כדי להצפין את המידע העובר דרך פרוטוקול TCP כדי שלא יהיה ניתן לראות אותו או לשכפל אותו ובכך לזייף בקשות.

לפי מודל ה-OSI פרוטוקול SSL שוכן בשכבת השיחה, בין שכבת התעבורה (4) לשכבת היישום. (7) בהקשר של פרוטוקולי האינטרנט המשמעות היא בטווח שבין TCP/IP-ל-HTTP, FTP, SMTP וכדומה SSL. אינו כולל מנגנון סינכרון מובנה ומסתמך על שכבת הקו שתחתיו SSL. מתחלק לשתי שכבות עיקריות, כמתואר בתרשים.

- **שכבת לחיצת יד** (Handshake layer) שמתחילה תהליך התקשרות, קובעת מספר פרמטרים משותפים כמו מספר גרסה ואלגוריתמים נתמכים. (cipher suits) במהלך לחיצת היד המשתתפים משלימים תהליך אימות זהויות ומייצרים את מפתח השיחה. שכבה זו מניחה כברירת מחדל מצב "צופן" NULL שמשמעו ללא הצפנה ואימות כלל.
- **שכבת רקורד** (Record layer) בשכבה זו מבוצעת חלוקת המידע העובר בקו לרשומות בגודל לכל היותר בתים. כאשר לכל רשומה צמוד תג אימות HMAC. שכבה זו תומכת בדחיסת נתונים, אם כי מסיבות של זכויות יוצרים לא מצוינת טכנולוגיה ספציפית למעט אלגוריתם NULL מנדטורי שאינו עושה דבר. בכך דחיסה הופכת לאופציה לא תיקנית תלוית מימוש.

בלחיצת היד משיגים שלושה יעדים:

1. הצדדים מסכמים ביניהם על "חבילת צופן", ערכה של אלגוריתמים קריפטוגרפיים (וגודל מפתחות ההצפנה) שייעשה בהם שימוש לצורך הפרוטוקול.
2. הצדדים מייצרים סוד משותף על מנת לגזור ממנו מפתחות שיחה.
3. הצדדים מאמתים את זהותם. אף על פי ש SSL-תומך באנונימיות, אימות חד-צדדי או אימות דו-צדדי, מקובל בשלב זה לוודא רק את זהות השרת.



התרשים מעל מתאר את הוריאציות השונות של פרוטוקול SSL. האות S מסמלת משלוח מסרים מאומתים מצד השרת C. מסמן אימות לקוח אופציונלי E. מייצג גרסה שבה משתמשים במפתחות ארעיים של פרוטוקול שיתוף מפתח דיפי-הלמן R. פירושן חידוש שיחה. צמד המסרים ChangeCipherSpec/Finished של הלקוח והשרת מתבצעים בסדר שונה אחד מהשני, אילו של השרת נשלחים מיד לאחר ServerHello.

- הלקוח פותח התקשרות על ידי משלוח בקשת ClientHello לשרת, המכילה פרטים מזיהום, סוג חבילת הצופן שנתמכת על ידו וכן מספר אקראי client\_random.
- בהתאם למידע שקיבל, השרת קובע את אופן ההתקשרות - סוג חבילת הצופן, מוסיף מספר אקראי משלו server\_random ושולח הודעת ServerHello בחזרה ללקוח. חילופי ההודעות נקראים "משא ומתן" (SSL negotiation). שני המספרים האקראיים של השרת והלקוח יהוו חלק מהסוד המשותף של השיחה הנוכחית.

- השרת מצרף כמו כן "תעודת מפתח ציבורי" (Public key certificate) X.509 "תואמת המכילה את זהותו והמפתח הציבורי שלו. בדרך כלל מדובר במפתח RSA, אם התצורה כוללת פרוטוקול שיתוף מפתח מצורפת תעודת DSS שבדרך כלל נושאת מפתח ציבורי ארוך טווח של פרוטוקול דיפי-הלמן.
- הלקוח מאמת את התעודה שקיבל מהשרת באמצעות תעודת השורש (root certificate) שבדפדפן שלו. אם קיים הבדל היררכי בין תעודת השורש שלו לבין תעודת השרת, השרת נדרש לשלוח רשימה מסודרת של כל התעודות הקשורות והלקוח יבדוק את תקפות שרשרת התעודות שקיבל עד שיתקל בתעודת השורש המובנית אצלו. בסיום שלב האימות נשלח המסר הריק, ServerHelloDone.
- הלקוח מייצר מספר אקראי חדש, pre\_master\_secret, מצפינו באמצעות המפתח הציבורי המאמת של השרת ושולח את התוצאה המוצפנת לשרת במסר ClientKeyExchange. נדרש בהתאם לכינון הנוכחי, הלקוח מייצר בנוסף מפתח ארעי (Ephemeral key) לצורך פרוטוקול דיפי-הלמן. אם ברשות הצדדים מפתח ציבורי ארוך טווח של דיפי-הלמן pre\_master\_secret, יהיה זהה בכל התקשרות.
- שני הצדדים גוזרים מפתח שיחה משותף בשני שלבים. תחילה משתמשים בפונקציה פסאודו-אקראית (SHA-2) לפי התקן הנוכחי) כדי לייצר ערך גיבוב מהפרמטרים: pre\_master\_secret, המספר client\_random והמספר server\_random. ערך גיבוב נקרא master\_secret. לאחר מכן חוזרים על פונקציית ה-PRF פעם נוספת ומייצרים ערך גיבוב מתוצאה האחרונה master\_secret יחד עם הערכים; המשפט "key expansion", המספר client\_random והמספר server\_random. התוצאה האחרונה נקראת "בלוק מפתח". בהתאם לחבילת הצופן שנבחרה (עם או בלי אימות), בלוק המפתח מחולק לשישה מקטעים (מפתחות); שני מפתחות הצפנה, שני וקטורי אתחול ושני מפתחות אימות, כל זוג עבור הלקוח והשרת בהתאמה.
- השרת יכול לדרוש אימות הלקוח. במקרה זה השרת שולח בקשת CertificateRequest המכילה את סוגי התעודות המותרות (כמו סוג אלגוריתם חתימה) ורשימה של רשויות מאשרות המזוהות בשמן המפורסם.
- הלקוח מגיב בהודעת Certificate המכילה תעודת מפתח ציבורי אחת או יותר וכן תמצית (Hash) של כל חילופי המסרים מלחיצת היד, כשהיא חתומה עם המפתח הפרטי המתאים לתעודה.
- אם פרוטוקול דיפי הלמן (DH) נבחר, אזי המפתחות הארעיים (Ephemeral keys) הציבוריים של שני הצדדים מוטמעים במסרים KeyExchange של כל אחד מהם בהתאמה. יחד עם המפתחות הפרטיים המתאימים, שני הצדדים מחלצים את הסוד

המשותף. `pre_master_secret` אך צריך לזכור שפרוטוקול דיפי-הלמן הבסיסי אינו מספק הגנה מפני התקפת אדם באמצע.

- שני הצדדים מסיימים את לחיצת היד על ידי שני מסרים נוספים `ChangeCipherSpec` המציין את המעבר לחבילת הצופן החדשה שהוחלט עליה במהלך לחיצת היד וכן הודעת `Finished` המציינת כי הם מוכנים לעבור למצב מוצפן ומכאן ואילך כל המסרים של שכבת הרקורד יהיו מוצפנים בהתאם.
- במקרה של שגיאה או אימות לא מוצלח, תשלח הודעת ההתראה `CloseAlert` והפרוטוקול והשיחה המשויכת יסתיימו מיד.
- הודעת חידוש מאפשרת לשני הצדדים לחדש את ההתקשרות בתהליך מקוצר. כשהשרת תומך בכך, הוא מצרף מחרוזת זיהוי שיחה להודעת `ServerHello`. במקרה זה הלקוח יכול לציין את מספר השיחה בבקשת ההתחברות שלו מבלי צורך לעבור את כל תהליך לחיצת היד עם האימות. אך עליו לצרף מספר אקראי חדש ולכן מפתח השיחה יהיה שונה בהתאם.

פרוטוקול זה משתמש ב הצפנה מסוג RSA:

RSA היא מערכת הצפנת מפתח ציבורי דטרמיניסטית מעשית הראשונה שהומצאה והיא עדיין בשימוש נרחב במערכות אבטחת מידע מודרניות, תקשורת מחשבים ומסחר אלקטרוני. ב-RSA, כבכל מערכת מפתח ציבורי, מפתח ההצפנה אינו סודי והוא שונה ממפתח הפענוח שנשמר בסוד, על כן היא נקראת אסימטרית. האסימטריה ב-RSA נובעת מהקושי המעשי שבפירוק לגורמים של מספר פריק שהוא כפולה של שני ראשוניים גדולים, שהיא בעיה פתוחה בתורת המספרים.

אליס משדרת לבוב את מפתחות ההצפנה  $(e, n)$  ושומרת בסוד את המפתח  $d$ . אם בוב מעוניין לשלוח את המסר  $m$  לאליס תחילה עליו להפוך את  $m$  לערך מספרי הנמוך מ- $n$  בדרך מוסכמת כלשהי.

$$c = m^e \bmod n$$

להצפנה בוב פשוט מעלה את  $m$  בחזקת  $e$  ונוטל את השארית מחילוק ב- $n$ . את הטקסט המוצפן  $c$  הוא יוכל לשדר לאליס בערוץ פתוח ונגיש לכל.

לפענוח אליס משחזרת את המסר  $m$  מתוך הטקסט המוצפן  $c$  בעזרת המפתח הסודי  $d$  בדרך זו:

$$m = c^d \bmod n$$

## HTTP

(Hypertext Transfer Protocol) HTTP: הוא פרוטוקול תקשורת שנועד להעברת דפי HTML ואובייקטים שהם מכילים (כמו תמונות, קובצי קול, סרטוני פלאש וכו') ברשת האינטרנט וברשתות אינטראנט. הפרוטוקול פועל בשכבת היישום של מודל ה-OSI ובשכבת היישום של מודל TCP/IP שרתי HTTPS הם שרתי התוכן המרכזיים ברשת האינטרנט ודפדפנים הם תוכנות הלקוח הנפוצות ביותר לפרוטוקול HTTPS.

התקשורת ב-HTTP מתחילה ביצירת שיחה בין השרת ללקוח באמצעות פרוטוקול TCP בשכבת התעבורה של פרוטוקול TCP/IP, ונמשכת בסדרה של בקשות (requests) ותשובות (responses) שנשלחות על ידי הלקוח והשרת, בהתאמה. ראשית, הלקוח יוצר חיבור לכתובת ה-IP ולפורט שבו השרת נמצא, בדרך כלל פורט 80. לאחר מכן נשלחת הבקשה, הכוללת את הכתובת של האובייקט המבוקש (למשל, דף HTML) ופרטים נוספים על הבקשה ועל הלקוח. השרת קורא את הבקשה, מפענח אותה, שולח ללקוח תשובה בהתאם ולרוב מנתק את החיבור ללקוח כשהשליחה הסתיימה.

מעצם הגדרתו, פרוטוקול HTTP הוא – stateless protocol – על מנת ליצור תקשורת בין הלקוח לשרת שמבוססת על היסטוריית הבקשות-תשובות בין השרת ללקוח נעשה שימוש בעוגיות (cookies). לדוגמה, שרת יכול לשתול במחשב הלקוח עוגייה עם אישור שהלקוח התחבר לחשבון מסוים עם סיסמה נכונה על מנת שלא יצטרך להקיש סיסמה בכל התחברות מחדש לאתר שמתארח על השרת.

בקשות HTTP מורכבות מהנתונים הבאים:

שיטת הבקשה.

כתובת של האובייקט המבוקש.

גרסת הפרוטוקול שלפיו מורכבת הבקשה.

שדות כותרת המתייחסים לבקשה, ללקוח, או לתוכן הנמצא בגוף הבקשה.

גוף הבקשה (אופציונלי).

נכון לגרסה 1.1, קיימות 8 שיטות בקשה: GET, HEAD, POST, PUT, DELETE, OPTIONS, TRACE ו-CONNECT. ניתן לסווג אותן בשני אופנים: שיטות בטוחות לעומת שיטות לא בטוחות, ושיטות אידמפוטנטיות (idempotent) לעומת שיטות שאינן אידמפוטנטיות. סיווגים אלו נועדו להוות אינדקציה כללית עבור שרתים, דפדפנים ובוני אתרים באשר למידת ההשפעה של כל אחת מהשיטות על השרת ועל הלקוח.

שיטות בטוחות מוגדרות ככאלה שמיועדות רק לשם קבלת מידע מהשרת; הן לא אמורות לשמש, למשל, לשליחת מידע מהלקוח לשרת, לביצוע שינויים כלשהם במסדי נתונים שנמצאים על השרת וכו'. במילים אחרות, אין לשיטות אלה תופעות לוואי, למשל תוצאות, פרט לקבלת מידע GET. ו-HEAD נחשבות לשיטות בטוחות.



שיטות אידמפוטנטיות הן שיטות ששליחה חוזרת שלהן גורמת בדיוק לאותן תוצאות כמו שליחה אחת. שיטות שהן בטוחות הן, לפי ההגדרה, גם אידמפוטנטיות, משום שאין להן כלל תופעות לוואי. השיטות PUT ו-DELETE גם הן נכללות בסיווג זה. במקרה של DELETE, למשל, בקשה נוספת למחיקה של אותו פריט תגרום לכך שהפריט לא ימצא על השרת. אפילו אם הוא כבר לא היה עליו אחרי בקשה אחת, התוצאה של הבקשות החוזרות היא בדיוק אותה תוצאה של הבקשה הראשונה.

אלו הן שיטות הבקשה הנתמכות בגרסה 1.1:

(GET קבל)

מיועדת לקבלת אובייקט שנמצא על השרת, בכתובת שניתנת בתחילת ההודעה. בקשות GET הן הנפוצות ביותר ברשת האינטרנט.

(HEAD כותרת)

מבקשת מהשרת לשלוח את כל שדות הכותרת שהיו נשלחים לבקשת GET, אך בלי האובייקט עצמו. השיטה נועדה, בין השאר, לאפשר בדיקה של קישורים שבורים או זמני שינויים של אובייקטים מבלי לבקש את כל האובייקט.

(POST שלח)

בקשות המכילות גוף הודעה. בקשות POST משמשות בדרך כלל לשליחה של מחרוזות לשרת. למשל מחרוזות ארוכות, כגון נתונים בטפסי HTML, לשם עיבוד.

(PUT שים)

מבקשת מהשרת לשמור את גוף ההודעה המצורף לבקשה בתור אובייקט, שכתובתו היא הכתובת שניתנה בתחילת הבקשה.

(DELETE מחק)

מבקשת מהשרת למחוק את האובייקט שכתובתו מצוינת בתחילת הבקשה.

(OPTIONS אפשרויות)

מבקשת מהשרת מידע על דרכי התקשורת האפשריות ביחס לאובייקט מסוים או ביחס לשרת עצמו.

לדוגמה: קריאה ל OPTIONS- יכולה להגיע עם תשובה מהשרת שהוא מקבל, GET, Post, PUT, Head ו OPTIONS- אבל לא מקבל DELETE ו TRACE-

( TRACE שחזר)

מבקש מהשרת לשלוח את הבקשה בדיוק כפי שקיבל אותה. הדבר שימושי לבדיקה של תחנות הביניים שנמצאות בין הלקוח לשרת ומעבדות את ההודעות העוברות דרכן.

( CONNECT התחבר)

הפרוטוקול אינו מגדיר את השימוש ב CONNECT- אך שומר את השיטה הזו לשימוש עבור שרתי פרוקסי שמסוגלים להתנהג כמו תעלות SSL.

( PATCH תיקון)

נועד על מנת לאפשר החלה של שינויים/תיקונים על אובייקט. בקשת PATCH מוגדרת כסט של פעולות לגבי איך לעדכן/לתקן אובייקט.

כל שרתי ה HTTP- הכלליים נדרשים ליישם לפחות את שיטות ה GET- וה HEAD- כל שיטה אחרת נחשבת אופציונלית.

כתובות בבקשות HTTP

כתובת בבקשת HTTP יכולה להיות כתובת מלאה (כמו: <http://www.w3.org/Protocols>) או כתובת יחסית לשרת (בדוגמה זו, /Protocols). השימוש בכתובות יחסיות הוא הנפוץ ביותר, דבר שהקשה בעבר על אחסון של מספר אתרים על אותו שרת, משום שהשרת לא יכול היה לדעת לאיזה אתר מבין האתרים המאוחסנים אצלו מכוונת הבקשה. כדי לפתור את הבעיה מבלי לאבד את התמיכה לאחור בגרסאות קודמות, גרסה 1.1 מחייבת כל בקשה לכלול שדה כותרת בשם host, שערכו הוא שם המתחם של האתר אליו מיועדת הבקשה.

## HTTPS

פרוטוקול HTTPS הוא אותו פרוטוקול כמו HTTP אך הוא מוצפן באמצעות SSL.

## תיאור מסכי מערכת

## מסך הרשמה

## תפקיד המסך

תפקיד המסך הזה הוא שהמשתמש יוכל להירשם למערכת באמצעות הכנסת שם משתמש וסיסמא.

## תיאור המסך

המסך מציג מקום בו ניתן להכניס שם משתמש ועוד מקום בו ניתן להכניס סיסמא כאשר בנוסף לכך יש כפתור שאפשר להירשם איתו כאשר לוחצים עליו הקוד בודק האם השם משתמש קיים במערכת והאם הסיסמא עונה על מספר קריטריונים. כאשר נוסף משתמש נשלח המידע על כך שנוסף משתמש לכל המחשבים. נוסף על כך יש כפתור שיוצר את המסך מחדש כך שאם נוצר משתמש בדיוק במחשב אחר אפשר ללחוץ על הכפתור והוא יעביר אותך לדף חסימת האתרים.



Guard Client

Username

Password

Setup Refresh

## מסך כניסה

### תפקיד המסך

תפקיד המסך הוא שהמשתמש יוכל להיכנס למערכת באמצעות השם משתמש והסיסמא שנכנסו איתם לפני.

### תיאור המסך

במסך יש מקום לשים שם משתמש וסיסמא כאשר יש כפתור להיכנס שהוא בודק האם המשתמש קיים ואם כן האם הסיסמא מתאימה למשתמש הזה אם כן הוא מעביר אותו לדף חסימת המשתמשים, בנוסף קיים כפתור היוצר את הדף מחדש כך שאם נמחק המשתמש הוא יעביר את המשתמש לדף ההרשמה.



The image shows a screenshot of a software window titled "Guard Client". The window has a light gray background and a title bar with standard window controls (minimize, maximize, close). Inside the window, there are two input fields: "Username" and "Password". Below these fields are two buttons: "Refresh" and "Login".

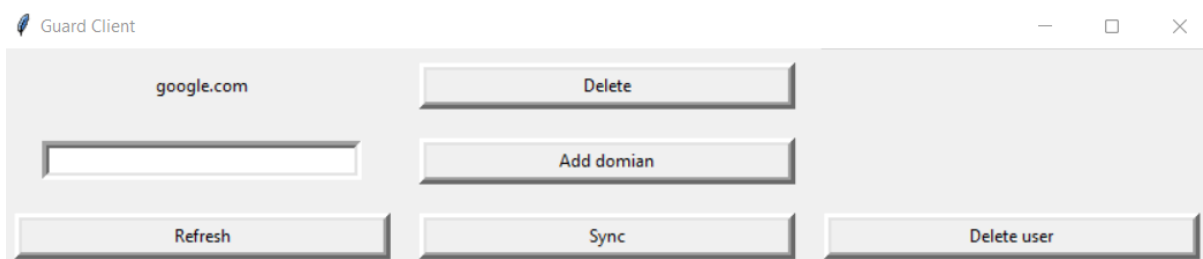
## מסך חסימת אתרים

## תפקיד המסך

להציג, להוסיף ולמחוק אתרים שחסומים.

## תיאור המסך

במסך מוצגים כל האתרים החסומים כאשר ליד כל אחד יש כפתור למחוק את האתר כאשר לוחצים על הכפתור הוא מוחק אותו ממסד הנתונים, מוחק אותו מקובץ HOST, ושולח לכל המחשבים על כך שנמחק DOMAIN. בנוסף לכך יש מקום לרשום DOMAIN כאשר לוחצים עליו הוא מוסיף את ה DOMAIN למסד הנתונים, מעדכן את קובץ HOST ו מעדכן כל המחשבים על כך שנוסף DOMAIN. יתר על כן ישנו כפתור של סכרון הנתונים שתפקידו לשלוח את כל המידע של מסד הנתונים לכל שאר המחשבים ובכך לסכרן את מסד הנתונים שלו עם של כולם. זאת ועוד קיים כפתור המעביר אותך לדף של מחיקת משתמש. וגם קיים כפתור היוצר את הדף מחדש ובכך אם היה שינוי במסד הנתונים בגלל שינוי שקרא במסך אחר הוא יופיע גם במסך הזה.



---

## מסך מחיקת משתמש

---

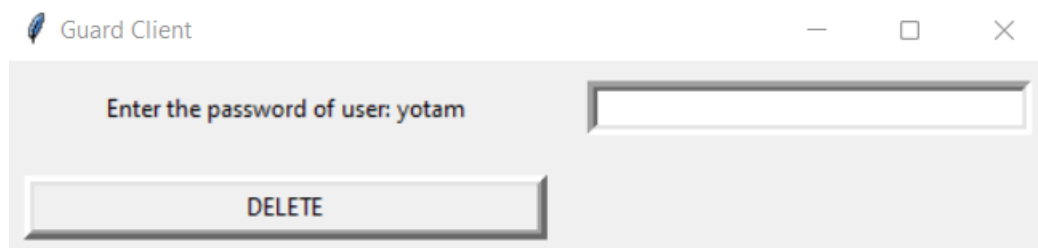
### תפקיד המסך

למחוק משתמש קיים

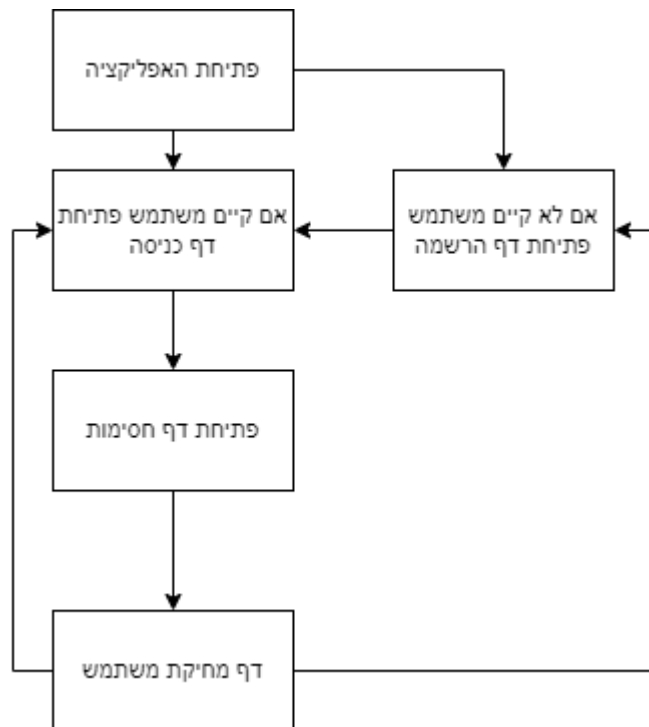
---

### תיאור המסך

במסך יש מקום לשים את הסיסמא וכפתור למחיקת הסיסמא כאשר לוחצים עליו הוא לוקח את הסיסמא על פי המשתמש ובודק האם הסיסמא המוצפנת שהוכנסה שווה לסיסמא ממסד הנתונים אם כן הוא מוחק את המשתמש ומעדכן את שאר המחשבים על כך אחר כך הוא פותח או את דף ההרשמה או את דף הכניסה דבר זה תלוי בהאם יש עוד משתמשים.



## היררכיית דפים



## תיאור מבני הנתונים

## מסד הנתונים הראשי (SQLITE)

מסד נתונים זה מכיל שתי טבלאות הראשונה שייכת לאחסון פרטי המשתמש כאשר שם כלול שם המשתמש כ TEXT מה שאומר שהוא מכיל טקסט שגודלו בלתי מוגבל, וכלול שם גם הסיסמא גם כן כ TEXT.

הטבלה השנייה במסד הנתונים הזה היא הטבלה שמכילה את האתרים הנחסמים כך שבטלה זו נכלל רק שדה אחד שהוא ה DOMAIN של האתר שרוצים לחסום

## קובץ HOST

בקובץ זה נכללים ה DOMAIN של האתרים ולא יזה IP ה DOMAIN הזה מוביל כך שאם מסתכלים עליו הוא נראה כך:

127.0.0.1 google.com

## סקירת חולשות ואיומים

### פיצוח סיסמא

האיומים היכולים לקרות על המערכת הם בעיקר ניסיון פיצוח סיסמא מכיוון שלא קיים מנגנון שמנוע ממך לנסות כמה סיסמאות שאתה רוצה אך כדי להתמודד עם כך הסיסמא חייבת להיות בעלת 8 תווים כאשר לפחות אחד מהם עם אות גדולה, אות קטנה, מספר וסימן ולכן כמות האפשרויות מאוד גדולה ולכן קשה מאוד לפרוץ לסיסמא בטח עם BRUTEFORCE. ובגלל שגם כל הסיסמאות מוצפנות באמצעות SHA-256 לכן לא ניתן לדעת את הסיסמא מבלי לנסות מלא פעמים.

### SQL INJECTION

יכול להיות שקיימים חולשות במסד הנתונים בגלל שאינני בודק האם קלט שמגיע מהמשתמש הוא שאלתת SQL אך אני לא מציג למשתמש נתונים ממסד הנתונים אלא אני רק מאכסן אותם ובודק אותם ואי אפשר להיכנס באמצעות הזרקת קוד SQL לקלט.

### DDOS

למערכת אכן ניתן לעשות DDOS אך לעשות דבר זה יכול להרוג את הנתב לפני שזה יהרוג את המערכת ולכן דבר זה בלתי סביר בעליל.

### ראיית המידע שעובר

כל המידע האישי שעובר, עובר דרך ערוצי תקשורת על TCP שהם מוצפנים באמצעות SSL מה שאומר שלא יהיה ניתן לראות את המידע שעובר אלא אם כן יש בידי הפורץ גישה לCERTIFICATE של המחשבים.

### הזרקת קוד

הזרקת קוד לא אפשרית בגלל שכל הקוד נמצא קובץ EXE או PYI שהם בלתי קריאים וקשה מאוד לשנות אותם כדי שיתנו את הפלט הרצוי.



## מימוש הפרויקט

### מודולים מיובאים

#### THREADING

מודל זה מאפשר לחלק את הקוד לתהליכונים וכלל את כל הדברים הדרושים על מנת להריץ אותם ולנהל אותם בצורה נכונה.

#### RE

מודול זה מאפשר שימוש בביטויים רגולריים בפיתון ובדיקת קלט מסוים כלפיהם. המודול מיועד על מנת לבדוק האם הסיסמא מכילה 8 תווים שכוללים לפחות אות גדולה, אות קטנה, מספר וסימן.

#### SOCKET

מודול האחראי לפתיחת ערוצי התקשורת והעברת המידע דרכם, משמש בכל ערוצי התקשורת גם TCP וגם UDP.

#### UUID

ספרייה המתעסקת בהבאת מזהים ייחודיים, משומשת על מנת להשיג את כתובת ה MAC של המחשב.

#### TYPING

ספרייה המשתמש לנתינת רמזים על הטיפוסים של כל משתמש ואיזה טיפוס הפונקציה מחזירה. משמש בכמעט כל פונקציה בפרויקט וכמעט כל משתנה בפרויקט.

#### CTYPES

ספרייה המאפשרת קריאה וביצועם של קבצי DLL. ספרייה זו משתמשת על מנת להריץ את הקוד כמנהל מערכת.

#### SYS

ספרייה זו מספקת פונקציות ומשתנים שונים המשמשים לתמרון חלקים שונים של סביבת הריצה של פיתון. ספרייה זו משתמש גם כן על מנת להריץ את הקוד כמנהל מערכת.

---

## NETIFACES

ספרייה המשתמש על מנת לספק ספירת ממשקי רשת ופירוטם במחשב. משומש בפרויקט על מנת לראות את כל כתובת ה IP של המחשב ובחירת ה IP הנכון.

---

## OPENSSL

ספרייה המשתמש ליצירת certificates באופן פשוט באמצעות כל מיני משתנים.

---

## SQLITE3

ספרייה המשתמש ליצירה ועריכתה של מסד הנתונים כך שהוא שיכלול את הנתונים הרצויים. משומש בכל הפרויקט על מנת ליצור ולערוך את מסד הנתונים.

---

## TKINTER

ספרייה המשמשת על מנת ליצור ממשק משתמש פשוט ונוח.

---

## LOGGING

ספרייה המשמשת לרישום נוח על מנת ליצור הודעות בקרה למערכת.

---

## SSL

ספרייה המשמשת על מנת להפוך את ערוצי התקשורת למוצפנים באמצעות פרוטוקול SSL.

---

## HASHLIB

ספרייה המשמשת על מנת להצפין מידע על ידי כל מיני אלגוריתמי הצפנות כמו SHA-256 או md5.

## מחלקות ומודלים כתובים

## COMPUTER

מחלקה זאת נועדה על מנת לייצג מחשב במערכת של המחלקה הם (המשתנים הרשומים כך הינם *(property)*:

*ip* - כתובת ה IP של המחשב הזה כמחרוזת.

*name* - השם של המחשב כמחרוזת.

*mac* - כתובת ה MAC של המחשב כמחרוזת.

*port* – הפורט שדרכו מתחברים לשרת של המחשב הזה כמספר שלם.

*server\_socket* – ערוץ התקשורת שפתוח בין השרת שנמצא על המחשב שמיוצג על ידי המחלקה לבין המחשב שאליו רץ הקוד.

*client\_socket* – ערוץ התקשורת שפתוח בין השרת שעליו רץ הקוד לבין המחשב שמיוצג על ידי המחלקה.

## HOSTCLIENT

מחלקה זו נועדה על מנת לייצג את קובץ HOST ולרכז את הגישה עליו דרך מחלקה אחת.

*default\_host\_file* – התוכן הראשוני שנמצא בקובץ HOST.

*lock* – בכדי למנוע בעיות עם כך שאי אפשר לגשת לקובץ HOST מכמה תהליכים בו זמנית.

## פונקציות

*remove\_enter* – פונקציה המקבלת רשימה של שורות כאשר בכל שורה יש "ח" חוץ מהאחרונה והיא מחזירה רשימה בלי הירידת שורה.

*get\_readlines\_from\_read* – פונקציה המקבלת טקסט ומחזירה רשימה של כל השורות יחד עם "ח" בסוף כל שורה.

*Domain\_in\_file* – פונקציה המקבלת את הטקסט של הקובץ ואת ה DOMAIN ובודק האם ה DOMAIN נמצא בטקסט.

*add\_domain* – פונקציה המקבלת DOMAIN וממוסיפה אותו לקובץ.

`remove_domain` – פונקציה המקבלת DOMAIN ומוחקת אותו מהקובץ

`return_to_default` – פונקציה המחזירה את הקובץ לתכנו הראשוני.

`open_write_file` – פונקציה המחזירה את הקובץ פתוח בצורה של לרשום

`open_read_file` – פונקציה המחזירה את הקובץ פתוח בצורה של קריאה.

## MULTISOCKET

מחלקה המייצגת את כל ניהול התקשורת בין המחשבים כאשר

`broadcast_address` – כתובת ה-BROADCAST של הנתב הזה.

`connected_computers` – מילון של כל המחשבים המחוברים כאשר המפתח הוא IP שלהם.

`receiving_socket` – ערוץ התקשורת של השרת במחשב הזה.

`broadcast_send_socket` – ערוץ התקשורת שדרכו שולחים את הודעות ה-BROADCAST.

`udp_server_socket` – ערוץ התקשורת שדרכו קוראים את הודעות ה-BROADCAST.

`computer` – האובייקט של המחשב המייצג את המחשב שעליו רץ הקוד.

### פונקציות

`open_socket` – פונקציה הפותחת את ההקשבה בערוץ התקשורת המשמש כשרת

`broadcast_message` – פונקציה המקבלת הודעת טקסט ושולחת אותו באמצעות ערוץ התקשורת האחראי לשליחת הודעות BROADCAST באמצעות UDP.

`sync_data` – פונקציה המקבלת את המסד נתונים ושולחת לכל המחשבים את הערכים ב DB על מנת שיסכרנו את עצמם איתו

`add_user` - פונקציה המקבלת DOMAIN ושולחת לכל המחשבים המחוברים שנוסף DOMAIN

`add_user` – פונקציה המקבלת שם משתמש וסיסמא מוצפנת ושולחת לכל המחשבים המחוברים שמשתמש נוסף יחד עם הערכים הללו

`remove_domain` – פונקציה המקבלת DOMAIN ושולחת לכל המחשבים המחוברים שנמחק DOMAIN.

remove\_user – פונקציה המקבלת שם משתמש ושולחת לכל שאר המחשבים על כך שנמחק משתמש.

## SQLCLIENT

מחלקה המייצגת את מסד הנתונים ורק דרכו מתנהל מסד הנתונים

db – משתנה המייצג את מסד הנתונים.

### פונקציות

add\_data\_to\_table – פונקציה המקבלת שם טבלה, טאפל של שמות טור, וטאפל של המידע הפונקציה מוסיפה את המידע במקומות המתאימים בטבלה המתאימה לפי הדברים שהפונקציה מקבלת.

update\_data\_from\_table – פונקציה המקבלת שם טבלה, שם טור לעדכן ומידע, בנוסף יכול להתקבל גם תנאי שכל השורות שענות על התנאי הזה יתעדכנו כך. הפונקציה מעדכנת את הנתונים במסד הנתונים.

delete\_data\_from\_table – פונקציה המקבלת שם טבלה, תנאי ומידע לאותו תנאי הפונקציה מוחקת את השורות הענות על התנאי.

create\_table – פונקציה המקבלת שם טבלה, שמות וסוגיהם של הטורים והאם שהטבלה תהיה עם מספר סידורי לכל שורה, הפונקציה יוצרת את הטבלה על פי הפרמטרים המתאימים.

delete\_user – פונקציה המקבלת שם משתמש ומוחקת את השורה עם השם המשתמש הנ"ל.

commit – פונקציה הדוחפת את השינויים למסד הנתונים.

get\_user – פונקציה המחזירה את השורה של פרטיו של המשתמש בהינתן הסיסמא המוצפנת שלו.

add\_user – פונקציה המקבלת שם משתמש וסיסמא ומוסיפה את המשתמש למסד הנתונים.

create\_tables – פונקציה היוצרת את כל הטבלאות עם הטורים המתאימים הדרושות לפרויקט.

get\_host\_rows – פונקציה המחזירה את כל הDOMAINS הנמצאים במסד הנתונים.

hash\_password – פונקציה המצפינה את הסיסמא.

get\_all\_users – פונקציה המחזירה את כל המשתמשים הקיימים במסד הנתונים.

get\_password – פונקציה המקבלת שם משתמש ומחזירה ממסד הנתונים את הסיסמא המוצפנת.

## BROADCAST

הינו קובץ המחיל את כל הדברים הנועדו על מנת לשלוח ולקבל בקשות BROADCAST.

### פונקציות

start\_broadcast\_setup – פונקציה המקבלת את מסד הנתונים, מנהל קובץ HOST ומנהל ערוצי התקשורת. הפונקציה פותחת שני תהליכים כאשר הראשון אחראי על שלוח הודעת BROADCAST כל 5 שניות והשני אחראי על להגיב בצורה נכונה להודעות שהוא מקבל.

broadcast – שולח את ההודעה who is up? בהודעת BROADCAST כל 5 שניות.

handle\_broadcast\_answer – פונקציה המקבלת כל הזמן הודעות BROADCAST ומחזירה תשובה מתאימה אם היא מקבלת who is up? היא תשמור את המחשב ששלח את ההודעה כאשר רק ה IP שלו ידוע ושולח הודעת תשובה מתאימה. אם מקבל את ההודעה קט אם כל הפרמטרים המתאימים הוא משנה/מוסיף את המחשב ששלח את ההודעה ויוצר איתו ערוץ תקשורת מאובטח.

handle\_client\_addition – פונקציה המקבלת מחשב המחובר ובודקת האם יש ערוץ תקשורת פתוח איתו אם כן אז הוא שולח לו הודעה ומוודא כך שהוא עדיין פועל כמו שצריך, אם הוא לא מצליח לשלוח הוא משנה לכך שאין לו ערוץ תקשורת פתוח איתו אם לא קיים ערוץ תקשורת הוא יוצר ערוץ תקשורת מאובטח על פי הנתונים שנישמרו במשתנה המחשב המחובר.

## GUICLIENT

מחלקה המייצגת את ה GUI של האפליקציה המחלקה מקבלת את מנהל ערוצי התקשורת, מנהל קובץ HOST ומנהל מסד הנתונים. הוא שומר בנוסף את כל האלמנטים הנמצאים בדף נוכחי על מנת לאפס את הדף בצורה נוחה.

### פונקציות

clear\_page – פונקציה ההורסת את כל האובייקטים בדף.

create\_sign\_up\_page – פונקציה היוצרת את דף ההרשמה.

sign\_up – פונקציה הלוקחת את הנתונים מדף ההרשמה ויוצרת משתמש אם הוא עונה על הקריטריונים ומעביר אותו אל דף הכניסה. אחרת נותנת הודעת שגיאה מתאימה

create\_login\_page – פונקציה היוצרת את דף הכניסה.

login – פונקציה הלוקחת את הנתונים מדף הכניסה ובודקת האם השם משתמש והסיסמא נכונים אם כן מעביר אותו לדף ניהול חסימות, אם לא מביא הודעת שגיאה מתאימה.

create\_host\_page – פונקציה היוצרת את הדף לניהול חסימות

refresh\_host\_page – מוחק את כל האובייקטים בדף ויוצרת את דף ניהול החסימות מחדש.

delete\_domain – מוחקת את הDOMAIN שאליו לחצו.

add\_domain – לוקח את הDOMAIN מהדף ומוסיף את ה DOMAIN למסד הנתונים. ובנוסף בובה את הדף מחדש.

delete\_user\_page – יוצר את דף מחיקת משתמשים.

delete\_user – בודק האם הסיסמא נכונה ומוחק את המשתמש.

refresh\_setup\_page – יוצר מחדש את דף ההרשמה.

## קוד לבעיות אלגוריתמיות

## מיון כתובות IP

```
def get_min(addr: List[Dict[str, str]], by: str):
    min_index = 0
    for i in range(len(addr)):
        minimum = addr[i][by]
        min_index = i
        for j in range(len(addr)):
            current_split = [int(y) for y in addr[j][by].split('.')]
            splitted_min = [int(x) for x in minimum.split('.')]
            if splitted_min[0] == current_split[0]:
                if splitted_min[1] == current_split[1]:
                    if splitted_min[2] == current_split[2]:
                        if splitted_min[3] > current_split[3]:
                            minimum = addr[j][by]
                            min_index = j
                    elif splitted_min[2] > current_split[2]:
                        minimum = addr[j][by]
                        min_index = j
                elif splitted_min[1] > current_split[1]:
                    minimum = addr[j][by]
                    min_index = j
            elif splitted_min[0] > current_split[0]:
                minimum = addr[j][by]
                min_index = j
    return min_index
```

```
def sort_addr(addr: List[Dict[str, str]], by: str):
    returned_list = []
    for i in range(len(addr)):
        index = get_min(addr, by)
        returned_list.append(addr[index])
        addr.remove(addr[index])
    return returned_list
```



## התמודדות עם תשובות BROADCAST

```
def handle_broadcast_answer(my_sockets: MultiSocket, sql_client: SQLClient, host_client: HostClient):
    while True:
        data, udp_addresses = my_sockets.udp_server_socket.recvfrom(1024)
        if udp_addresses[0] != my_sockets.computer.ip:
            decoded_data = data.decode()
            globals.Logger.info(f"RECEIVED BROADCAST MESSAGE {decoded_data} from {udp_addresses}")
            splitted_data = decoded_data.split(',')
            if splitted_data[-1] == "up":
                connected_computer = Computer(ip=splitted_data[0], subnet_mask=splitted_data[1], mac=splitted_data[2],
                                                port=int(splitted_data[3]), name=splitted_data[4])
                if connected_computer.ip in my_sockets.connected_computers:
                    my_sockets.connected_computers[splitted_data[0]].update_computer(connected_computer)
                    threading.Thread(target=handle_client_addition,
                                    args=(my_sockets.connected_computers[splitted_data[0]],)).start()
                else:
                    my_sockets.connected_computers[splitted_data[0]] = connected_computer
                    threading.Thread(target=handle_client_addition,
                                    args=(my_sockets.connected_computers[splitted_data[0]],)).start()
            elif splitted_data[-1] == "who is up":
                if udp_addresses[0] not in my_sockets.connected_computers:
                    my_sockets.connected_computers[udp_addresses[0]] = Computer(ip=udp_addresses[0])
                    my_sockets.broadcast_message(
                        f"{my_sockets.computer.ip},{my_sockets.computer.subnet_mask},{my_sockets.computer.mac},{my_sockets.computer.port},{my_sockets.computer.name},up")
```

## התמודדות עם הודעות של CLIENTS

```
def handle_connections(my_sockets: MultiSocket, client_socket: SSLSocket, client_address: Tuple[str, int],
                       sql_client: SQLClient, host_client: HostClient):
    if client_address[0] not in my_sockets.connected_computers:
        my_sockets.connected_computers[client_address[0]] = Computer(ip=client_address[0], client_socket=client_socket)
        connected_computer = my_sockets.connected_computers[client_address[0]]
    else:
        connected_computer = my_sockets.connected_computers[client_address[0]]
        connected_computer.client_socket = client_socket

    while True:
        received_data = client_socket.recv(1024)
        if received_data != b'':
            decoded_data = received_data.decode()
            logging.info(f"got {decoded_data} from {connected_computer}")
            if decoded_data.startswith("add domain"):
                splited_decoded_data = decoded_data.split(" ")
                sql_client.add_data_to_table("host", ("domain",), (splited_decoded_data[2],))
                host_client.add_domain(splited_decoded_data[2])
            elif decoded_data.startswith("remove domain"):
                splited_decoded_data = decoded_data.split(" ")
                sql_client.delete_data_from_table("host", where="where domain=?", data=(splited_decoded_data[2],))
                host_client.remove_domain(splited_decoded_data[2])
            elif decoded_data.startswith("add user"):
                try:
                    splited_decoded_data = decoded_data.split(" ")
                    sql_client.add_user(splited_decoded_data[2], splited_decoded_data[3])
                except sqlite3.IntegrityError:
```

```
        pass
    except Exception as e:
        raise e

    elif decoded_data.startswith("remove user"):
        try:
            splited_decoded_data = decoded_data.split(" ")
            sql_client.delete_user(splited_decoded_data[2])
        except sqlite3.IntegrityError:
            pass
    except Exception as e:
        raise e
```

## הירשמות לאפליקציה

```
def sign_up(self, username_entry: ttk.Entry, password_entry: ttk.Entry):
    error_label = ttk.Label(self.root, width=35)
    self.elements.append(error_label)
    if username_entry.get() != '':
        if " " not in username_entry.get():
            if password_entry.get() != '':
                if re.compile(r'^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$').match(
                    password_entry.get()) and " " not in password_entry.get():
                    if self.sql_client.get_data_from_table(table_name='users', where="where username=?",
                                                            variables=(username_entry.get(),), amount_to_fetch=1,
                                                            data_to_select="password") is None:
                        username = username_entry.get()
                        self.sql_client.add_user(username_entry.get(), password_entry.get())
                        self.multi_socket.add_user(username_entry.get(), password_entry.get())
                        self.clear_page()
                        self.create_host_page(username)
                    else:
                        error_label.configure(text="Username already in usage")
                else:
                    error_label.config(width=150)
                    error_label.configure(
                        text="Password has to be more than 8 characters and have at list one digit,one lowercase "
                        "letter, one uppercase letter and one symbol with no space")
            else:
                error_label.configure(text="No password")
```

```
        else:
            error_label.configure(text="Username has to contian no spaces")
    else:
        error_label.configure(text="No username")
    try:
        if error_label['text'] != '':
            error_label.grid(row=2, column=1, columnspan=10, padx=10, pady=10)
    except Exception:
        pass
```

## מסמך בדיקות

## WIRESHARK

על מנת לבדוק האם הפקטות נשלחות כהלכה ניתן לפתוח wireshark ולראות האם בקשות ה-BROADCAST עוברות כהלכה מבלי בעיות בנוסף לכך ניתן לראות שפקטות ה-TCP עוברות כהלכה אך אי אפשר לראות את תוכנם, אם לא רואים את הפקטות עוברות מומלץ לכבות את ה-FIREWALL במערכת ההפעלה.

## DB BROWSER

על מנת לוודא כי המסד נתונים מתעדכן כהלכה ניתן להוריד אפליקציה בשם DB BROWSER כאשר איתה ניתן לראות את הנתונים במסד הנתונים, אם ומסד הנתונים לא מתעדכן צריך לוודא כי מסד הנתונים הוא אכן בשם הנכון ("database.db") ואם הוא לא בשם זה יש לשנות אותו.

## האם קובץ HOST משתנה

על מנת לוודא כי החסימות מתנהלות כמו שצריך ניתן לבדוק האם קובץ HOST משתנה לערכים הנכונים הערכים הנכונים האמורים להיות זה ה DOMAIN ומיד אחריו הכתובת הבאה 127.0.0.1 אם הוא לא מתעדכן כמו שצריך ניתן לעשות כמה דברים:

1. לוודא כי המיקום של קובץ HOST במחשב הוא כמו בקוד  
(C:\Windows\System32\drivers\etc\host)
2. לאפס את הקובץ למצבו ההתחלתי (ניתן לעשות זאת בקלות באמצעות האובייקט  
(HostManagment)

## לבדוק האם החסימה פועלת

להיכנס לאתר שנחסם ולוודא כי האתר אכן נחסם. אם האתר לא נחסם ניתן לבדוק האם יש אפליקציה אחרת אשר משנה את הקובץ HOST ובנוסף מומלץ לעשות את שני הצעדים הבאים:

1. לרשום ב CMD את הפקודה הבאה "ipconfig /flushdns" כדי למחוק את המטמון DNS של המחשב ובכך להכריח את המחשב לעבור דרך קובץ HOST.
2. ב CHROME ללכת לכתובת הבאה <chrome://net-internals/#dns> וללחוץ על clear host cache. בכך בעצם מוחקים גם את המטמון DNS של CHROME.

## מסמך LOGS

על מנת לוודא כי כל התקשורת עוברת נכון וכי הקוד עובד כמו שצריך מומלץ לפתוח את קובץ ה LOG כאשר שם מודפס כל הודעה שנכנסת ויוצאת בנוסף למילון של כל המחשבים המחוברים ובכך אפשר לראות האם כל המחשבים מחוברים עם ערוצי התקשורת הנכונים ועם הנתונים הנכונים.

אם אחד הנתונים לא נכונים אפשר לבדוק כמה דברים:

1. האם ה IP נכון? למחשב יש כל מיני IP כאשר הקוד לוקח את ה IP השני בסדר אם הוא אינו ה IP השני בסדר יש לשנות את ה INDEX מהרשימה.

2. האם הPORT נכון יש לוודא כי פורט 4000 ופורט 5000 פנויים ואם לא אז לשנות בקוד כך שהם יהיו פנויים.

## מדריך למשתמש

## עץ קבצים

```
admin.py
client.crt
client.key
constants.py
database.db
desktop.ini
globals.py
logfile.log
main.py
main.spec
server.crt
server.key
SocketHandler.py
ssl_generation.py
__init__.py

+---.idea
|   .gitignore
|   misc.xml
|   vcs.xml
|   workspace.xml
|
|   \---inspectionProfiles
|       Project_Default.xml
|
+---GUI
|   GUIClient.py
|
|   \---__pycache__
|       GUIClient.cpython-310.pyc
|
+---HostFileManagment
|   HostManagment.py
|
|   \---__pycache__
|       HostManagment.cpython-310.pyc
|
+---NetworkTalk
|   Computer.py
|   desktop.ini
|   MultiSocket.py
|   __init__.py
|
|   \---__pycache__
|       Computer.cpython-310.pyc
|       desktop.ini
|       MultiSocket.cpython-310.pyc
|       __init__.cpython-310.pyc
```

```

+---SQLManagment
|   SQLClient.py
|
|   \---__pycache__
|           SQLAlchemy.cpython-310.pyc
|
+---StartUp
|   broadcasting.py
|   desktop.ini
|   discovery.py
|
|   \---__pycache__
|           broadcasting.cpython-310.pyc
|           desktop.ini
|
\---WebApp
|   https_app.py
|   http_app.py
|
|   \---__pycache__
|           https_app.cpython-310.pyc
|           http_app.cpython-310.pyc

```

## התקנת מערכת

על מנת להתקין את המערכת דרושים כמה דברים:

### סביבת עבודה

על מנת לעבוד על הקוד דרוש פייטון של 3.10 ומעלה ודרושים כל הספריות הפורטו במודלים מיובאים.

### הכלים הנדרשים

אין כלי ספציפי הדרוש על מנת לעבוד על הקוד או להריץ אותו אך כן מומלץ להשתמש ב PYCHARM או ב VSCODE על מנת לקרוא את המידע ב מסד הנתונים אני ממליץ להשתמש ב DB BROWSER.

### רשת

על מנת שהפרויקט יעבוד דרוש דברים מאוד מינימליים על מנת שהחסימות יעבדו בעוד מחשבים חוץ ממחשב אחד והמחשבים יסונכרנו דרוש שיהיה נתב.

## מדריך לניווט חלונות למשתמש

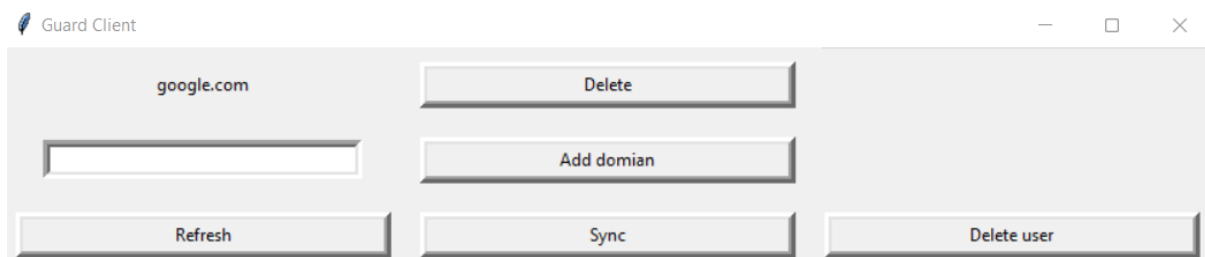
כשפותחים את האפליקציה בפעם הראשונה במחשב הראשון ברשת שאליו מותקן הקוד רואים את הדבר הבא



The screenshot shows the 'Guard Client' window. It contains two input fields: 'Username' and 'Password'. Below these fields are two buttons: 'Setup' and 'Refresh'.

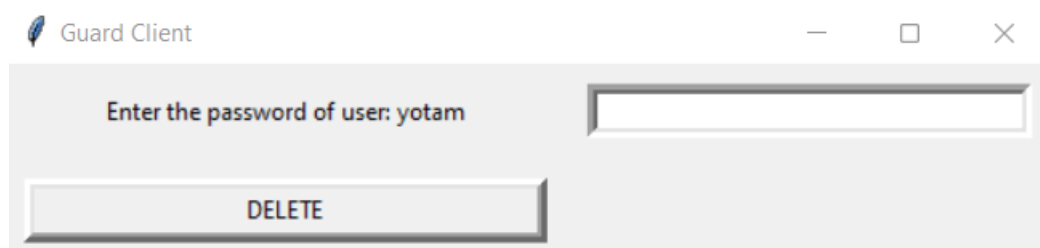
במסך זה שמים את שם משתמש והסיסמא המיועדים לרשת זו יש לשים לב כי הסיסמא כוללת לפחות אות גדולה, אות קטנה, תו מיוחד, ומספר וגודלה מעל 8 תווים. על מנת להמשיך לדף הבא ולהירשם יש ללחוץ על כפתור ה setup.

המסך הבא יהיה מסך חסימת אתרים



The screenshot shows the 'Guard Client' window with domain management options. It displays 'google.com' with a 'Delete' button. Below this is an input field and an 'Add domain' button. At the bottom are buttons for 'Refresh', 'Sync', and 'Delete user'.

הוא יופיע ריק (בלי ה google.com) על מנת להוסיף חסימה יש לרשום את הכתובת של האתר בלי www וללחוץ על הכפתור של Add Domain על מנת למחוק DOMAIN יש ללחוץ על הכפתור DELETE ליד ה DOMAIN שרוצים למחוק, אם נוצרו שינויים ומחשב אחר לא קלט אותם מסיבה מסוימת כל מה שצריך לעשות זה ללחוץ על כפתור ה SYNC שיסכרן את המידע של המחשב הזה עם כל שאר המחשבים. אם רוצים למחוק את המשתמש יש ללחוץ על DELETE USER שיוביל לדף הבא:



The screenshot shows the 'Guard Client' window with a password confirmation prompt: 'Enter the password of user: yotam'. There is an input field for the password and a 'DELETE' button below it.



על מנת למחוק את המשתמש כל מה שצריך זה לשים את הסיסמא של המשתמש וללחוץ על DELETE.

אם קיים כבר משתמש יופיע המסך הבא:



יש לשים את שם המשתמש וסיסמא של המשתמש וללחוץ על הLOGIN וזה יעביר לדף החסימות שמוסבר למעלה.

## רפלקציה

במהלך הפרויקט נתקלתי באתגרים קשים מאוד כמו לסכרן את המחשבים להצפין את המידע ועוד בנוסף לכך היו הצלחות בניהם לבנות GUI שנראה טוב לבנות מערכת נוחה למסד הנתונים ולקובץ HOST ולכן שיהיה קוד קריא מאוד. מלבד לפיתון בסיסי שנלמד בבית ספר כל השאר הייתי צריך ללמוד לבד וכך עשיתי את כל ה SQLITE ואת המידע הרלוונטי על ההיררכיה של DNS בנוסף לכך למדתי איך להשתמש ב GITHUB על מנת ליצור היסטוריה נוחה של הקוד. הכלים שאני לוקח איתי הם איך להשתמש ב GITHUB ואיך לנהל מסד נתונים בנוסף איך ליצור ערוצי תקשורת מאובטחים. קיבלתי עזרה מכל הצוות המקצועי בבית ספר ומעמיתי בכיתה בנוסף לכך עזרתי לעמיתי גם כן, בראייה לאחור ישנם חלקים בפרויקט שהייתי עושה אחרת כמו נעילת קובץ HOST כך שאי אפשר לשנות אותו ובנוסף לגרום לכך שה GUI והסרבר הם שתי אפליקציות שונות שיכולות לקשר אחת עם השנייה. אני רוצה להודות לכל הצוות המקצועי על התמיכה הרבה בין אם זה לענות לשאלות ובין אם זה להיות איתנו בכיתה למשך שעות רבות.

## ביבליוגרפיה

[/https://www.wikipedia.org](https://www.wikipedia.org) - ויקיפדיה

[/https://stackoverflow.com](https://stackoverflow.com) - Stack overflow

[/https://flask.palletsprojects.com/en/2.1.x](https://flask.palletsprojects.com/en/2.1.x) - Flask

[/https://docs.python.org](https://docs.python.org) - הספריות הנכללות בתוך פייטון

[/https://www.youtube.com](https://www.youtube.com) - יוטיוב

[/https://www.w3schools.com](https://www.w3schools.com)

[/https://github.com](https://github.com)

מורי וחברי לכיתה